Organ Clinic Data Base

INDEX:

Group Composition and Purpose:

Group: 05

Members: Lorenzo García-Herraiz, Elisa Melgosa, Carlo Bottini

Topic : Organ Clinic

Reasons for the program:

This program seeks to make for a more appropriate, comfortable, flexible and improved communication system between a patient and doctor. To be able to do this, the doctor will have most of the control, whereby he will be able to add patients, delete, see all information related to the patients operation and the patient will be able to see all his data to check for any errors.

User Stories :

- As patient i want to see my account profile
- As a patient I want to be able to modify my profile.
- As a patient I want to be able to log in.
- As a patient I want to be able to log out

- As a Doctor I want to be able to log in.
- As a Doctor I want to be able to log out.
- As a Doctor I want to be able to look for the best post-operatory treatment available.
- As a Doctor I want to be able to know how many nurses are available.
- As a Doctor I want to be able to change a patient's status on his clinical history
- As a Doctor I want to be able to see a patient's clinical history
- As a Doctor I want to be able to change the phase of the transplant process of my patient.
- As a Doctor I want to be able to know the compatibility of the patient with the organ (type blood, tamaño del organo, compatible con al edad)
- As a Doctor I want to be able to know how many of each organ there are available.
- As a Doctor I want to be able to know the characteristics of the organ that would be donated.

Use Cases:

(Doctor Use Cases)

- Actor: Doctor
- Name/interaction: Look the characteristics of an organ
- Goal: The Doctor can see the characteristics of an organ that is selected
- Preconditions:
    - A Doctor must exists.
    - The patient must exists
    - An organ must exists
- Standard path: The Doctor looks for the characteristics
- Postcondition: The Doctor can see the characteristics of the selected organ
- Alternate path: none

- Actor: Doctor
- Name/interaction: Look the compatibility of the patient with the organ
- Goal: The Doctor can see if the organ matches with the patient
- Preconditions:
    - A Doctor must exists.
    - The patient must exists
    - An organ must exists
- Standard path: The Doctor looks for the compatibility
- Postcondition: The Doctor knows if the organ is compatible
- Alternate path:
    - The organ is not compatible

- Actor: Doctor
- Name/interaction: Look the clinical history of a patient
- Goal: The Doctor can see the clinical history from a patient
- Preconditions:
    - A Doctor must exists.
    - A patient must exists.
- Standard path: The Doctor see the clinical history
- Postcondition: The Doctor knows the clinical history information of the patient.
- Alternate path: none.

- Actor: Doctor
- Name/interaction: Change patient's status
- Goal: The Doctor can modify the patient's status of his clinical history
- Preconditions:
    - A Doctor must exists.

    - A patient must exists
- Standard path: The Doctor modifies the status
- Postcondition: The status of the patient is modified
- Alternate path:
    - The status doesn't need to be changed, so the doctor exits from the option.

- Actor: Doctor
- Name/interaction: Asign nurse to operation
- Goal: The Doctor knows which nurse is available and asigns
- Preconditions:
    - A Doctor must exists.
    - A nurse must exists
- Standard path: The Doctor can see which nurse is available
- Postcondition: The Doctor knows which nurse is available or not
- Alternate path: none

- Actor: Doctor
- Name/interaction: Look for the best treatment
- Goal: The Doctor can choose the best treatment for the patient
- Preconditions:
    - A Doctor must exists.
    - The patient must exists
    - The patient must have the conditions to look for a treatment
    - There must have 1 or more treatment available
- Standard path: The Doctor choose the best treatment that matches with the patient
- Postcondition: The treatment for the patient is chosen.
- Alternate path:
    - There is no available treatment

- Actor: Doctor
- Name/interaction: Log out
- Goal: The Doctor exit from the system
- Preconditions:
    - A Doctor must exists.
    - The Doctor must had enter into the system correctly
- Standard path: The Doctor exits from the system
- Postcondition: The Doctor is not anymore in the system
- Alternate path: none

- Actor: Doctor
- Name/interaction: Log in
- Goal: The Doctor enter into the system
- Preconditions:
    - A Doctor must exists.
    - The User and Password must be correct
- Standard path: The Doctor enters into the system and can interact with it.
- Postcondition: The Doctor is in the system with his user.
- Alternate path:
    - The Doctor password is not correct: It notifies the system and permit to try again.
    - The Doctor User introduced is not correct: It notifies the system and permit to try again.

- Actor: Doctor
- Name/interaction: Add patient
- Goal: The Doctor enters a patient into his list of patients
- Preconditions:
    - Doctor must exist.
    - Patient must exist
    -
- Standard path: The Doctor enters into the system and can interact with it to enter the patient
- Postcondition: The Doctor has 1 more patient
- Alternate path:
    - The Patient does not exist. Repeat the nameto check if it is the right name. try again notified by the system.

(Patients Use Cases)

As a patient I want to be able to log in.

- Actor: Patient
- Name/Interaction: login
- Goal: log into the application
- Preconditions:
    - must be a patient(exist)
- Standard path: the patient logs in
- Postcondition: patient is in the system
- Alternate path: repeat asking for login if incorrect

As a patient I want to be able to log out.

- Actor: Patient
- Name/Interaction: log-out
- Goal: log out the application
- Preconditions:
    - must be a patient(exist)
- Standard path: the patient logs in
- Postcondition: patient is in the system
- Alternate path: repeat asking for login if incorrect

As a patient I want to be able to modify my account details

- Actor: Patient
- Name/Interaction: Patient modifies account's details
- Goal: modify my own details
- Preconditions:
    - must be logged in
- Standard path: modify case
- Postcondition: account details modified
- Alternate path: none

As patient i want to see my account profile

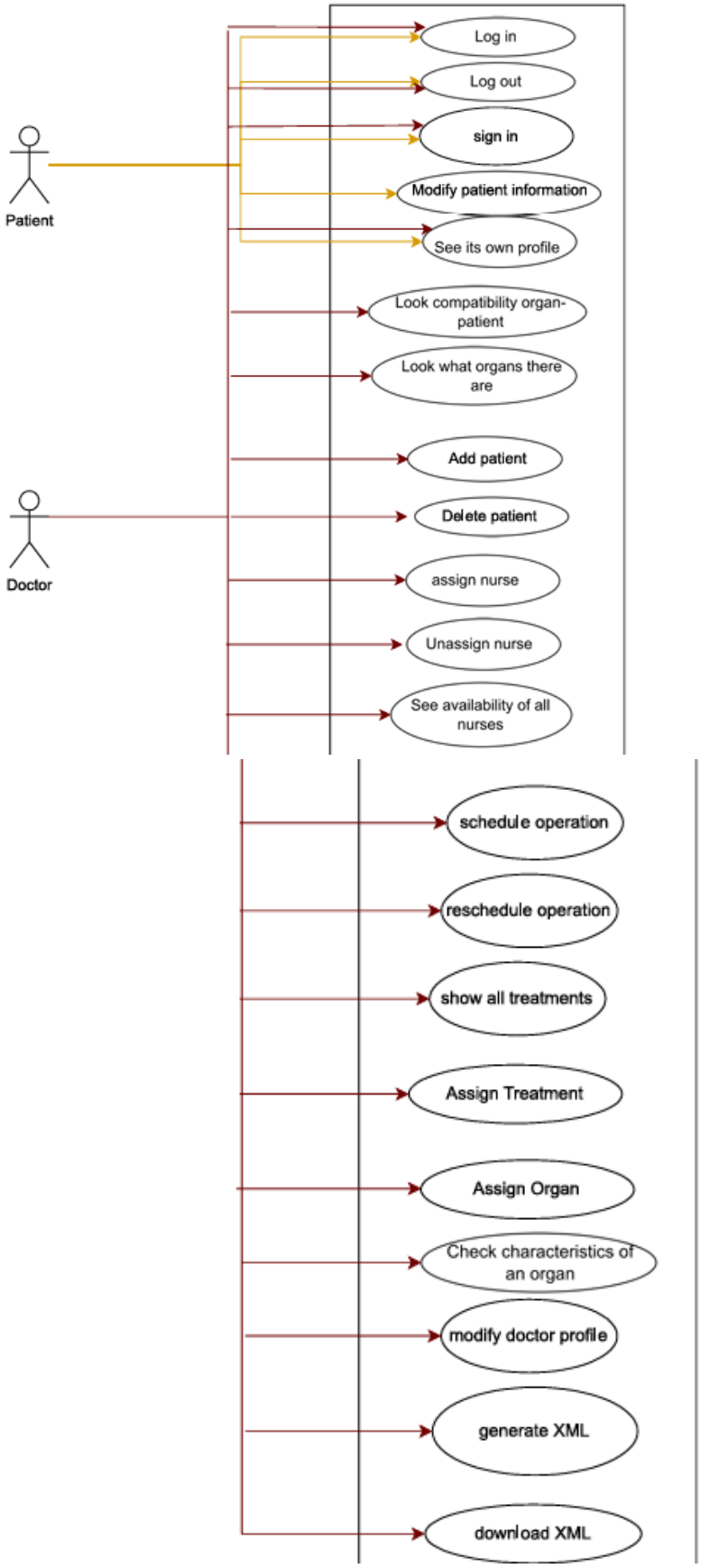
- Actor: Patient
- Name/Interaction: account info
- Goal: see my own account details
- Preconditions:
    - must be a patient(exist)
    - logged in
- Standard path: option to see information
- Postcondition: view info
- Alternate path: none

(Nurse Use Cases)

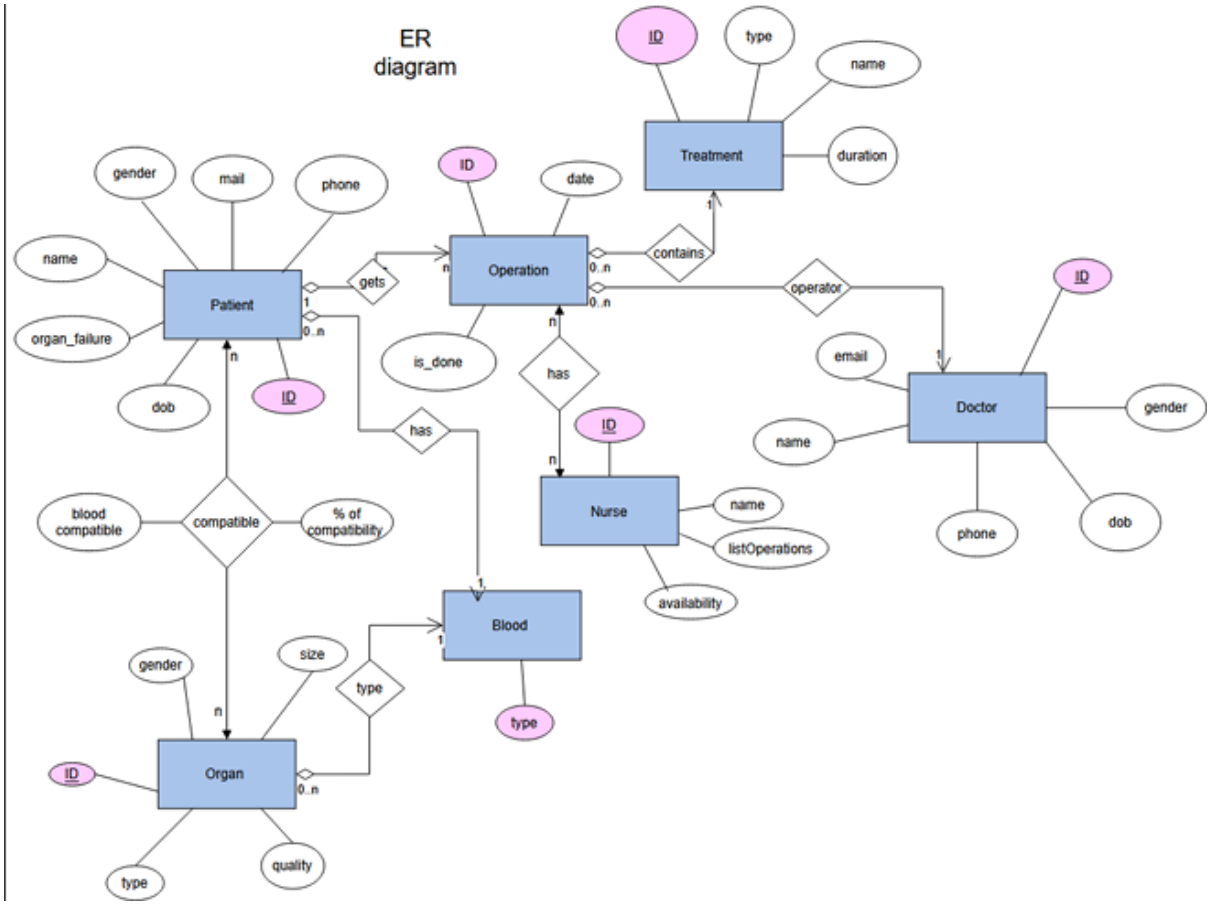
As a nurse i want to be available for the doctor in the surgery

- Actor: Nurse
- Name/Interaction: be available for the doctor in the surgery
- Goal: be available for the doctor in the surgery
- Preconditions:
    - the doctor and the nurse exists
- Standard path: be available
- Alternate path:not being available
- Postcondition: do the surgery

Use Case Diagram:



E-R Diagram

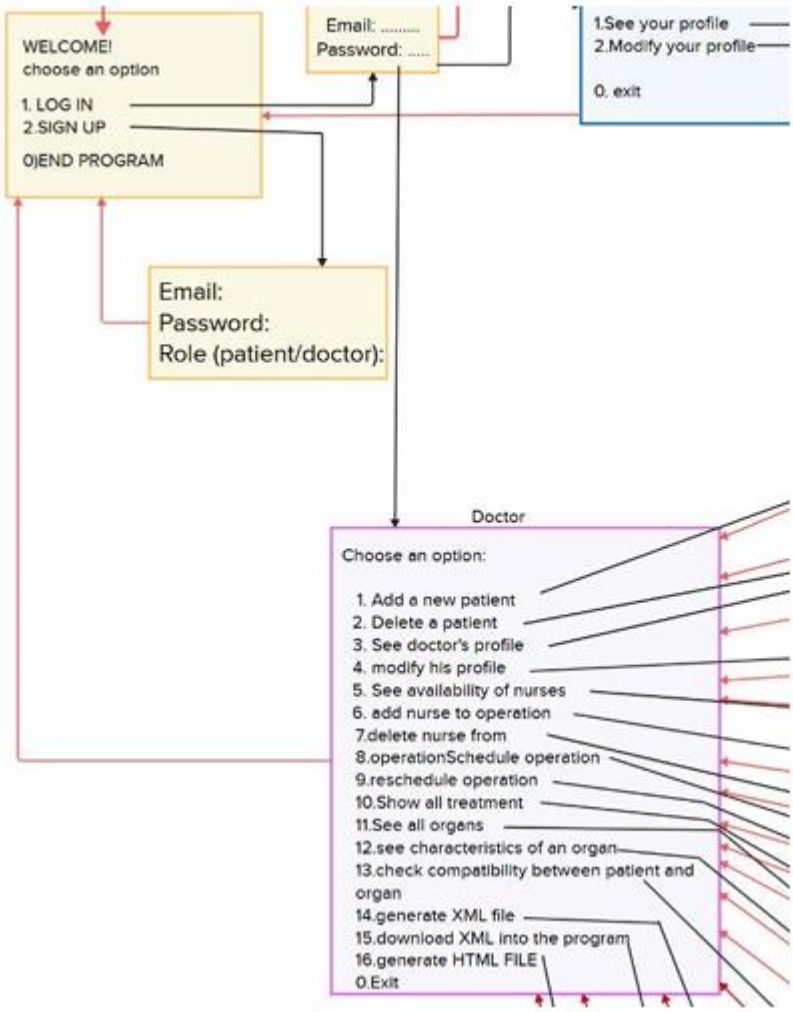

ER
diagram
ID
type
name
Treatment
duration
gender
mail
phone
ID
date
name
Patient
gets
Operation
contains
1
n
0..n
0..n
operator
ID
organ_failure
is_done
has
email
1
Doctor
gender
dob
ID
n
ID
name
has
n
blood
compatible
compatible
% of
compatibility
Nurse
name
listOperations
phone
dob
availability
1
Blood
1
gender
size
type
type
n
ID
Organ
0..n
type
quality

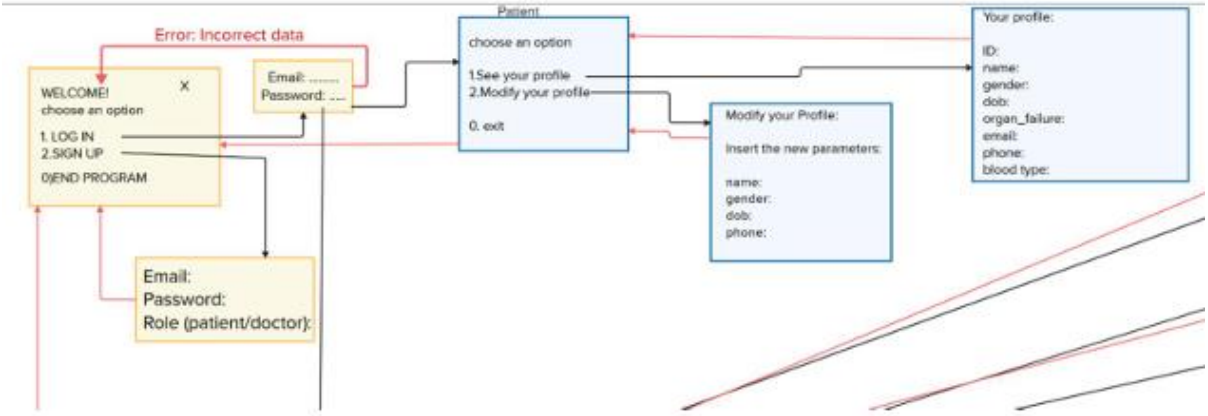
Tracability Matrix

Functional
The patient can see the doctors information(F1)
patient can see his account profile(F2)
the patient can modify his profile(F3)
The patient can log in(F4)
The patient can log out (F5)
The patient can reschedule appointment (F6)
The doctor can look all organs (F7)
The doctor can look for compatibility organ-patient (F8)
The doctor can look for the patient profile (F9)
The doctor can add a patient(F10)
The doctor can check characteristics of an organ (F11)
The doctor can look for a treatment(F12)
The doctor can add a nurse to a operation (F13)
The doctor can delete nurse from a operation (F14)
Not functional
The systems will complete the command in less than 2 seconds(NF1)
The system is programmed in Java (NF2)
The same patient can't be register twice(NF3)
The patient only has one doctor (NF4)
The same organ cannot be registered twice (NF5)

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	NF1	NF2	NF3	NF4	NF5
See if the operation has gone well									X						X	X			
Add a new operation				X						X					X	X			
Make an appointment	X			X		X			X	X					X	X		X	
See if the organ is compatible							X	X	X		X				X	X			X
Modify a patient		X	X	X	X										X	X	X		
Add patient										X					X	X	X		
Delete a patient															X	X	X		
Add nurse													X		X	X			
Delete nurse														X	X	X			
See his profile	X								X						X	X		X	
Diagnose Treatment								X				X			X	X			X
See availability						X									X	X			

Mock-Up


WELCOME!
choose an option
1. LOG IN
2.SIGN UP
0)END PROGRAM
Email: .....
Password: .....
1.See your profile
2.Modify your profile
0. exit
Email:
Password:
Role (patient/doctor):
Doctor
Choose an option:
1. Add a new patient
2. Delete a patient
3. See doctor's profile
4. modify his profile
5. See availability of nurses
6. add nurse to operation
7.delete nurse from
8.operationSchedule operation
9.reschedule operation
10.Show all treatment
11.See all organs
12.see characteristics of an organ
13.check compatibility between patient and organ
14.generate XML file
15.download XML into the program
16.generate HTML FILE
0.Exit



Fill patients data:
1.NAME
2.SURNAME
3.DOB
4.GENDER
5.ORGAN FAILURE
6.EMAIL
7.TELEPHON
8.BLOOD TYPE

You will delete a patient from the data:
1. INTRODUCE THE PATIENT ID .....

Profile of the Doctor:
name:
id:
dob:
gender:
email:
phone:

Modify your Profile:

Insert the new parameters:
name:
dob:
gender:
phone:

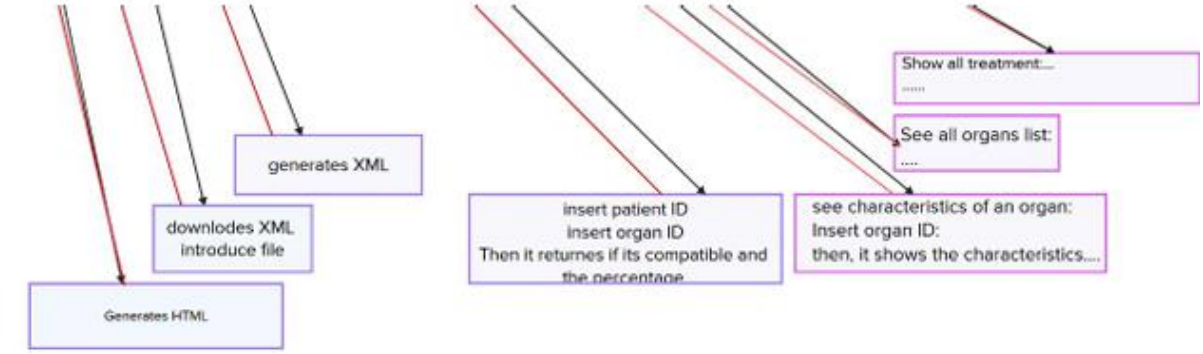
Nurses and their availability:
Ferran Torres: T
Luis gaarcia: T
Andrea garcia: F
...

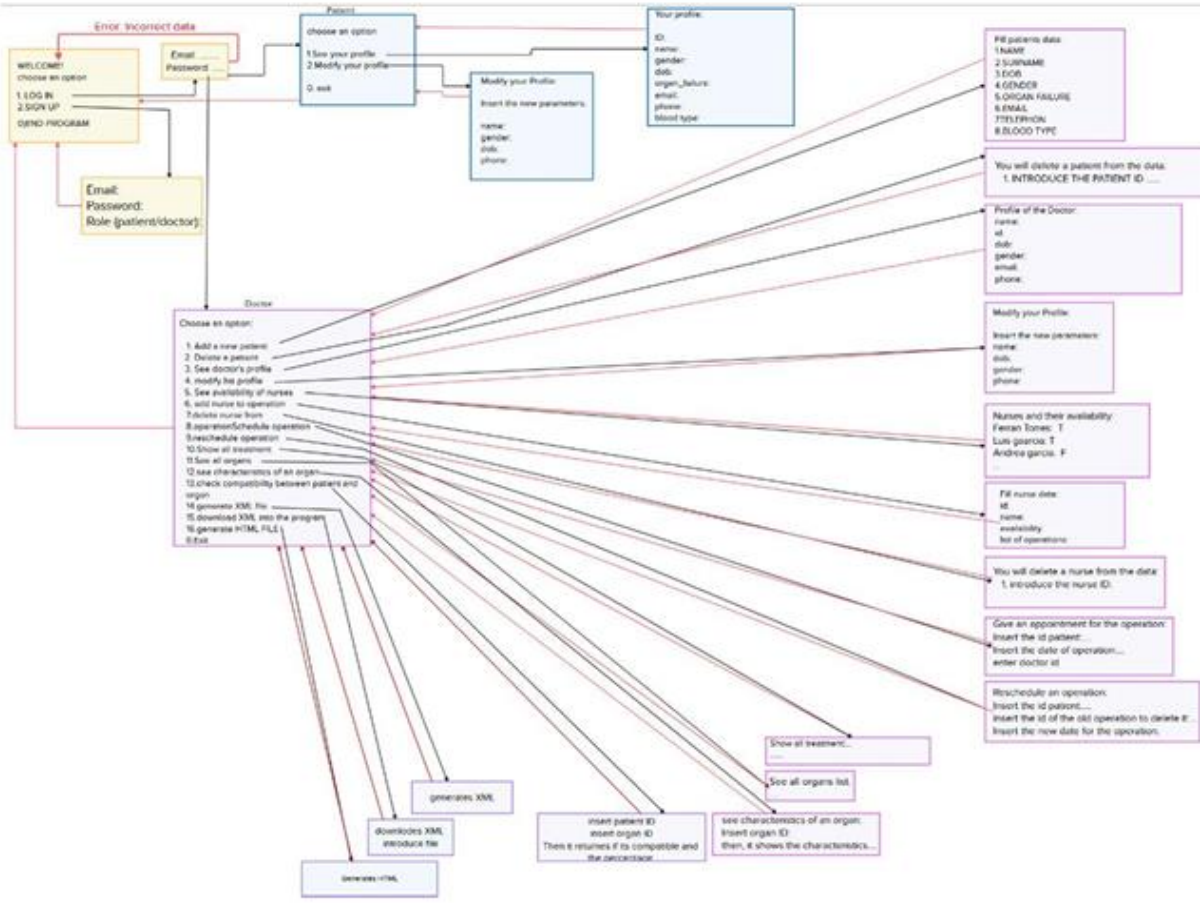
Fill nurse data:
id:
name:
availability:
list of operations:

You will delete a nurse from the data:
1. introduce the nurse ID:

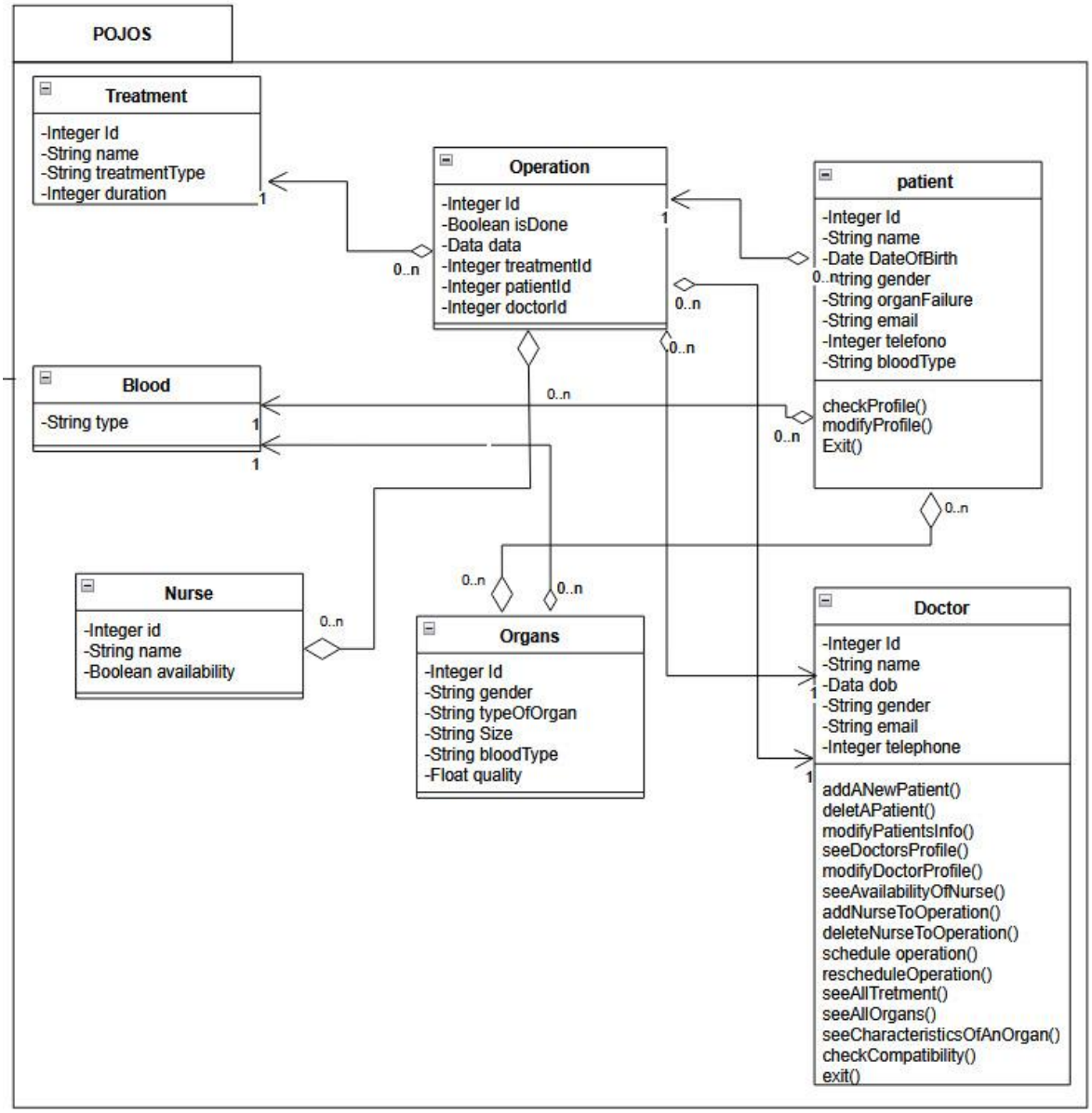
Give an appointment for the operation:
Insert the id patient:....
Insert the date of operation:...
enter doctor id

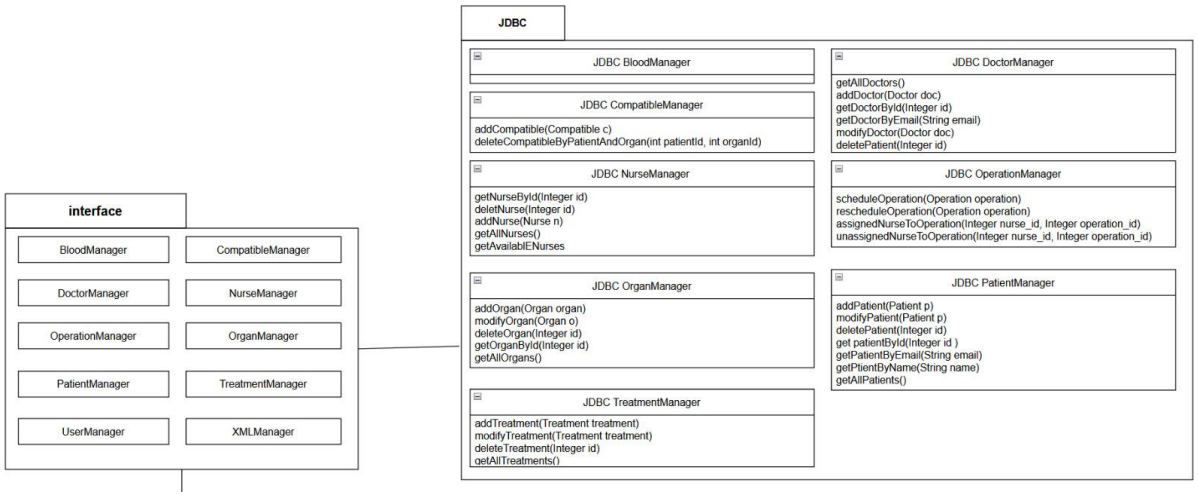
Reschedule an operation:
Insert the id patient:....
insert the id of the old operation to delete it:...
Insert the new date for the operation:

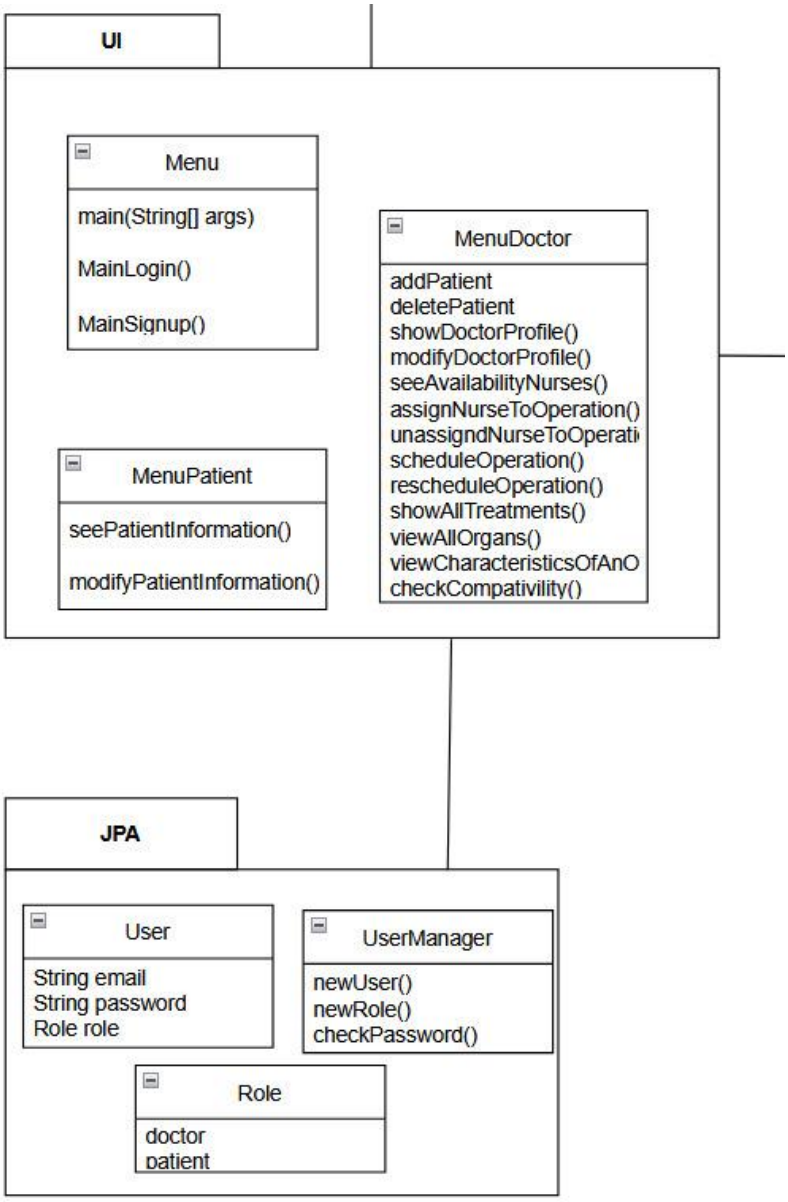

generates XML
downlodes XML
introduce file
Generates HTML
insert patient ID
insert organ ID
Then it returnes if its compatible and
the percentage
Show all treatment:...
.....
See all organs list:
....
see characteristics of an organ:
Insert organ ID:
then, it shows the characteristics....


Error. Incorrect data
WELCOME!
choose an option
1.LOG IN
2.SIGN UP
0)END PROGRAM
Email
Password
Email:
Password:
Role (patient/doctor):
Patient
choose an option
1.See your profile
2.Modify your profile
0. exit
Your profile:
ID:
name:
gender:
dob:
organ_failure:
email:
phone:
blood type:
Modify your Profile:
Insert the new parameters:
name:
gender:
dob:
phone:
Doctor
Choose an option:
1. Add a new patient
2. Delete a patient
3. See doctor's profile
4. modify his profile
5. See availability of nurses
6. add nurse to operation
7.delete nurse from
8.operationSchedule operation
9.reschedule operation
10.Show all treatment
11.See all organs
12.see characteristics of an organ
13.check compatibility between patient and
organ
14.generate XML file
15.download XML into the program
16.generate HTML FILE
0.Exit
Fill patients data:
1.NAME
2.SURNAME
3.DOB
4.GENDER
5.ORGAN FAILURE
6.EMAIL
7TELEPHON
8.BLOOD TYPE
You will delete a patient from the data:
1. INTRODUCE THE PATIENT ID
Profile of the Doctor:
name:
id:
dob:
gender:
email:
phone:
Modify your Profile:
Insert the new parameters:
name:
dob:
gender:
phone:
Nurses and their availability:
Ferran Torres: T
Luis garcia: T
Andrea garcia: F
Fill nurse data:
id:
name:
availability:
list of operations
You will delete a nurse from the data:
1. introduce the nurse ID:
Give an appointment for the operation:
Insert the id patient:
Insert the date of operation:
enter doctor id
Reschedule an operation:
Insert the id patient:
Insert the id of the old operation to delete it:
Insert the new date for the operation:
Show all treatment:
See all organs list
see characteristics of an organ:
Insert organ ID:
then, it shows the characteristics...
generates XML
downlodes XML
introduce file
insert patient ID
insert organ ID
Then it returnes if its compatible and
the percentage

UML


POJOS
Treatment
-Integer Id
-String name
-String treatmentType
-Integer duration
Operation
-Integer Id
-Boolean isDone
-Data data
-Integer treatmentId
-Integer patientId
-Integer doctorId
patient
-Integer Id
-String name
-Date DateOfBirth
-String gender
-String organFailure
-String email
-Integer telefono
-String bloodType
checkProfile()
modifyProfile()
Exit()
Blood
-String type
Nurse
-Integer id
-String name
-Boolean availability
Organs
-Integer Id
-String gender
-String typeOfOrgan
-String Size
-String bloodType
-Float quality
Doctor
-Integer Id
-String name
-Data dob
-String gender
-String email
-Integer telephone
addANewPatient()
deletAPatient()
modifyPatientsInfo()
seeDoctorsProfile()
modifyDoctorProfile()
seeAvailabilityOfNurse()
addNurseToOperation()
deleteNurseToOperation()
schedule operation()
rescheduleOperation()
seeAllTretment()
seeAllOrgans()
seeCharacteristicsOfAnOrgan()
checkCompatibility()
exit()
1
0..n


interface
BloodManager
CompatibleManager
DoctorManager
NurseManager
OperationManager
OrganManager
PatientManager
TreatmentManager
UserManager
XMLManager
JDBC
JDBC BloodManager
JDBC CompatibleManager
addCompatible(Compatible c)
deleteCompatibleByPatientAndOrgan(int patientId, int organId)
JDBC NurseManager
getNurseById(Integer id)
deletNurse(Integer id)
addNurse(Nurse n)
getAllNurses()
getAvailablENurses
JDBC OrganManager
addOrgan(Organ organ)
modifyOrgan(Organ o)
deleteOrgan(Integer id)
getOrganById(Integer id)
getAllOrgans()
JDBC TreatmentManager
addTreatment(Treatment treatment)
modifyTreatment(Treatment treatment)
deleteTreatment(Integer id)
getAllTreatments()
JDBC DoctorManager
getAllDoctors()
addDoctor(Doctor doc)
getDoctorById(Integer id)
getDoctorByEmail(String email)
modifyDoctor(Doctor doc)
deletePatient(Integer id)
JDBC OperationManager
scheduleOperation(Operation operation)
rescheduleOperation(Operation operation)
assignedNurseToOperation(Integer nurse_id, Integer operation_id)
unassignedNurseToOperation(Integer nurse_id, Integer operation_id)
JDBC PatientManager
addPatient(Patient p)
modifyPatient(Patient p)
deletePatient(Integer id)
get patientById(Integer id)
getPatientByEmail(String email)
getPtientByName(String name)
getAllPatients()



XML

In our project, XML was utilized to enable structured data storage and efficient communication between the system , especially for handling patient and doctor information. We employed JAXB (Java Architecture for XML Binding) to transform Java objects into XML format using a marshaller, and to convert XML files back into Java objects using an unmarshaller.

To perform marshalling, the patient2XML and doctor2XML methods convert Patient and Doctor instances into XML files. This is done so by creating a JAXBContext for the respective class, followed by the use of a Marshaller to serialize the object data into a properly formatted XML file. These XML files are stored in the ./xmls directory, and the XML output is also displayed in the console for verification.

XSLT

To make our project look better and easier to present, we used XSLT (Extensible Stylesheet Language Transformation) to turn the XML files into HTML pages. This made it easier to keep the data consistent, andhelped keep a clear separation between how data is stored and how it is seen.

The methods `patient2Html` and `doctor2Html` are in charge of converting XML files into HTML using XSLT. First, the object is turned into XML. That transformer processes the XML and generates an HTML file, which is saved in the `./xmls` folder.

DTD

To wrap things up, we added a DTD (Document Type Definition) for our two XML files. This helps keep the data accurate and consistent by setting clear rules and structure for each file. Thanks to the DTD, the XML can be validated to make sure it follows the expected format, which helps keep things standardized and makes it easier for different systems to work with the same data.